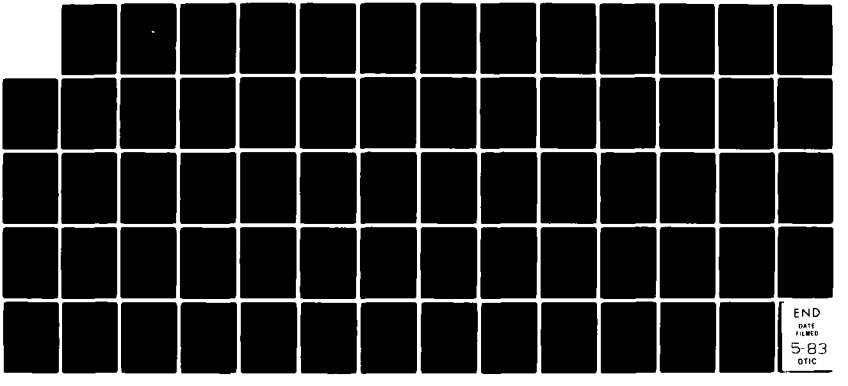
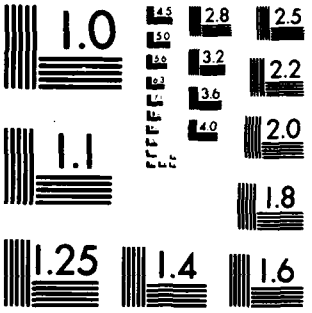
AD-A127 164 INTEGRATION CONSIDERATIONS FOR THE STOCK POINT LOGISTICS INTEGRATED COMMU..(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA K M BARRETT DEC 82 1/1

UNCLASSIFIED F/G 9/5 NL

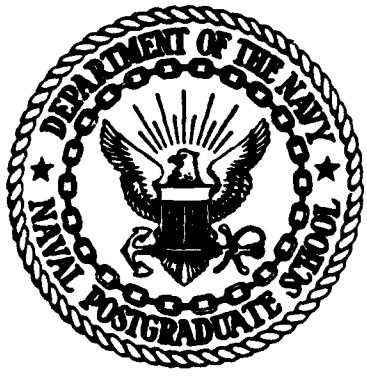


END
DATE FILMED
5-83
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

NAVAL POSTGRADUATE SCHOOL
Monterey, California



THESIS

INTEGRATION CONSIDERATIONS FOR THE STOCK POINT LOGISTICS INTEGRATED COMMUNICATIONS ENVIRONMENT (SPLICE) LOCAL AREA NETWORK

by

Kathleen M. Barrett
December, 1982

Thesis Advisor: Norman F. Schneidewind

Approved for Public Release; Distribution Unlimited

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A129 164 / 3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Integration Considerations for the Stock Point Logistics Integrated Communications Environment (SPLICE) Local Area Network	5. TYPE OF REPORT & PERIOD COVERED Master's Thesis December, 1982
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Kathleen M. Barrett	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940	12. REPORT DATE December, 1982
	13. NUMBER OF PAGES 66
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for Public Release; Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
SPLICE, LAN, DDN, TCP, Datagram, Virtual Circuit, Functional Module, Addressing

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis examines the various functional modules that have been designed in support of the Stock Point Logistics Integrated Communications Environment (SPLICE) local computer network. Initially, the overall design methodology is presented, followed by a description of the functional modules, their proposed capabilities and their relationships to each other. Finally, an analysis is made to determine how well the modules (continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT (Continued) Block # 20

fit together to form an operational local computer network and to support both inter- and intra-network communications.

Approved for public release; distribution unlimited.

Integration Considerations for the Stock Point Logistics Integrated Communications Environment (SPLICE) Local Area Network.

by

Kathleen M. Barrett
Lieutenant Commander, United States Navy
B.A., Immaculata College, 1972

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
December 1982

Author: Kathleen M. Barrett

Approved by: Norman F. Schneidewind
Thesis Advisor

Norman R. Lyon
Second Reader

Chairman, Department of Administrative Sciences

Dean of Information and Policy Sciences

ABSTRACT

This thesis examines the various functional modules that have been designed in support of the Stock Point Logistics Integrated Communications Environment (SPLICE) local computer network. Initially, the overall design methodology is presented, followed by a description of the functional modules, their proposed capabilities and their relationships to each other. Finally, an analysis is made to determine how well the modules fit together to form an operational local computer network and to support both inter- and intra-network communications.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

A. GENERAL OVERVIEW

The information contained in this section was obtained from a series of reference documents produced by both Naval Supply Systems Command (NAVSUP) and Fleet Material Support Office (FMSO) and is included as background [Refs.1,2,3].

The Stock Point Logistics Integrated Communications Environment (SPLICE) project is being developed to augment the existing Navy Stock Point and Inventory Control Point ADP facilities that support the Uniform Automated Data Processing System-Stock Points (UADPS-SP).

Presently, there are twenty new applications systems in various stages of development which will require a considerable amount of interactive processing and telecommunications support. The current UADPS-SP hardware is a Burroughs Medium Size (B-3500/3700/4700/4800) System, which will not be able to support these requirements without a total redesign effort and possible mainframe replacement. To support the interactive and telecommunications capabilities required, individual project managers for the new applications systems development will be utilizing a variety of minicomputers. These systems will all be implemented within the next four to five years according to projected schedules.

The development of SPLICE will have two major impacts. It will meet the increased need for the use of CRT display terminals to interact with application logic and retrieve information from the system data base and it will also address the need for a standard teleprocessing hardware and software environment to support the many new projects that

will impact all Navy UADPS-SP sites. This standardization will provide major economic benefits in the stages of design, development, operation and maintenance which will occur at approximately sixty geographically distributed sites, each having a different mix of application and terminal requirements.

At this time, the SPLICE processors are planned to be co-located with the host Burroughs Medium System at each Stock Point (SP) activity and with the Burroughs and Univac systems at the Inventory Control Points (ICP's). The SPLICE project proposes a distinct division of processing responsibilities called a "foreground/background" concept. The SPLICE foreground, utilizing a standard minicomputer hardware and software set, will serve as a front-end processor for the Burroughs system via a Local Area Network (LAN) interface and will handle communication lines as well as support the interactive operations. This interactive transaction processing support will be accomplished using the Terminal Applications Processing System (TAPS) data communication terminal management for both on-line and host processing terminals, and networking communication logic for Navy-wide distributed and satellite activity capability. On the host background processor (initially the Burroughs Medium System computers at each Stock Point), the functions performed will include large volume updating of master files, creating hard copy reports and other functions not requiring interactive immediate response access.

B. THESIS OBJECTIVES

A portion of the SPLICE project was initiated at the Naval Postgraduate School, Monterey, California, at the request of FMSO, to develop and design suggested alternatives for SPLICE Local Area Networks. The purpose of

this thesis is to examine these alternate design specifications and indicate what integration considerations have been met and those areas that remain to be addressed among the various functional modules that have been developed in support of both intranetwork communications occurring within the LAN itself and internetwork communications between the LAN and the Defense Data Network (See Appendix C).

C. BACKGROUND

The implementation of SPLICE as proposed by NAVSUP reflects a tightly coupled architecture which utilizes a centralized "complex manager" concept. The complex manager performs all required coordination between the SPLICE system components, or functions. Eight of these software functions have been identified for development to support the SPLICE concept. These functions are Terminal Management, Terminal Applications, Data Set Management, Peripheral Management, Batch Applications, Complex Management, Stock Point Front-End Processor Support and Stock Point Host-Resident Support. The first six functions will provide an application independent environment for LAN processing, while the last two support the Stock Point Host interface for foreground/background communication. It should be noted here that the entire SPLICE design, as outlined in the SPLICE System Specifications [Ref. 2] and the SPLICE Software Design [Ref. 1] revolves around a predetermined desire to use the Terminal Applications Processing System (TAPS), which is currently in existence at various Navy Stock Points. TAPS is designed to provide Communications Management (CM), Application Management (AM) and Data Management (DM) capabilities necessary for the Navy application systems to support on-line interactive query and update processing on the foreground complex.

The alternate SPLICE functional design approach taken here at the Naval Postgraduate School is directed towards designing the logical or virtual Local Area Network first, thus ensuring that functional requirements are satisfied [Ref. 4]. This is accomplished through the development of Functional Modules and their characteristics and the determination of communication protocols necessary to support them. The need for a complex manager is eliminated by providing the necessary control structures for a fully distributed LAN. The ability to move a functional module from one functional node to another will provide higher system availability than in the case of fixed assignment of functional modules to physical nodes. Through the proper distribution of functional resources across the nodes of the LAN, the failure of any one node would be transparent to the user and a higher degree of overall LAN reliability is provided than would exist with the use of a centralized complex manager.

II. LOCAL AREA NETWORK AND FUNCTIONAL MODULE OVERVIEW

A. LOCAL AREA NETWORK (LAN) DESIGN

A LAN has been defined as a data communication network, typically a packet communication network, limited in geographical scope [Ref. 5].

Basically, local area networks provide for the interconnection of data processing and computing devices located within a limited geographical area. They are primarily aimed at providing a communications means for processes resident in the multiple hosts which connect to the network. LANs have been installed and implemented in various forms for a multitude of functions, and have experienced varying degrees of success [Refs. 6, 7].

Due to the future increases of computing devices at the Navy's Supply Points and Inventory Control Points, networking of the devices within the local area offers the potential to efficiently share available resources. A local network structure capable of providing compatible interconnection of various terminals, data processing devices, word processors, gateways to other computer networks and of virtually any type of digital communication device, can provide an extremely flexible and highly reliable environment for SPLICE configurations.

A major advantage of local area networks in general is that, once implemented, the local area network can support practically any type of system transition. Another significant aspect of a well-designed local network is that it can support a long-term, vendor-independent transition strategy.

Local networks do create certain problems that must be considered, however. Provisions must be made for speed matching between the local area network and the long-haul network. In this particular instance, this matching must occur between the SPLICE LAN and the DDN. It can reasonably be presumed that the LAN will have a much higher data rate than the DDN. Thus, when a large number of packets are sent into the LAN to reach their ultimate destination through the DDN, packets may arrive at the gateway much faster than the gateway is able to pass them to the DDN. A mechanism is required to prevent the gateway from exhausting its buffer space. Additionally, the virtual circuit protocol in the LAN must be compatible with that of the DDN to allow for easy translation.

One of the basic elements which one must consider when dealing with LAN design is the topology method used for network interconnection. This issue is an important one in the performance of the LAN and is presented more fully in the following section.

B. BUS TOPOLOGY

Network topology is the arrangement of digital devices, called nodes, relative to the interconnecting media. In the recent evolution of local computer networks, several topologies have emerged. The SPLICE reliability requirements (as outlined in the Functional Description) preclude a system in which the network can be made inoperative by a single component failure. SPLICE configurations will also be subject to changes over time. Thus, SPLICE requires a topology which provides high resilience to single component failure while also allowing for system growth and reconfiguration. For these reasons, a bus architecture was chosen [Ref. 8].

The global bus configuration is an interconnection scheme in which all network computers or nodes are party to a single communications channel which is used in a message or packet switching mode. The channel may be a single wire pair or coaxial cable or even a multi-wire conduit. All node-to-node communications takes place over this shared channel. The channel operates in a broadcast multiple access mode, similar to that of an internal computer bus, where a transmission by any of the nodes can be received by all of the remaining nodes in the network. Access to the channel is controlled by any of a number of different time multiplexing schemes. The global bus topology for local networks has several inherent advantages over other topologies, including low cost and ease of incremental network expansion. Throughput and message delay characteristics are highly dependent on the access control protocol used [Ref. 9]. In this shared channel computer communication network, only one message can be transmitted on the channel at a time. Thus, it must be determined who can transmit at any given time via some distributed control mechanism. Additionally, an addressing mechanism is required to aid in data flow patterns. As will be seen in a later diagram (Figure 2.1), the logical design of the SPLICE LAN provides two types of buses: a data bus which will transfer the actual applications messages and a control bus which will carry administrative traffic (such as resource allocation and error messages). Few local networks available from vendors, however, provide separate data and control buses due to the additional cost and hardware required. Thus, physical implementation of this design will utilize one bus [Ref. 4]. This physical configuration can be seen in Figure 2.2 .

C. FUNCTIONAL MODULES

A functional module is defined as one which provides a generalized function for many applications. As opposed to having many sets of applications modules, there will exist one set of functional modules which can provide services for many applications. This design methodology was chosen to save time and money in system development and implementation [Ref. 10]. Since many functions are common to various applications, a great deal of duplicate work in the areas of system analysis, design, programming, coding, testing and maintenance will result if applications modules are designed and implemented for each and every application. The functional module approach will eliminate this redundancy and also result in the better utilization of computer resources, primarily memory and file storage space. This last benefit is due to the fact that the major differences in applications usually exist in the input and output formats and applications parameters; they are not normally in the basic operations of editing data, maintaining files and generating displays and reports.

In the Postgraduate School design of the LAN, the generalized approach using functional modules is emphasized. Ten functional modules have been identified to date, although not all have been completely designed. They are listed in Table I .

These modules are divided into two basic categories: operating functions such as the transaction processing modules, and support functions which are those that exist to make effective use of the processing modules and the entire LAN. Figure 2.1 illustrates the logical connections as envisioned in the SPLICE LAN design. The actual configuration is envisioned as being similar to that shown in Figure 2.2 . It illustrates a LAN configuration utilizing three minicomputers in addition to the mainframes at SPLICE sites.

TABLE I

Functional Modules

Local Communications
National Communications
Front-End Processing
Terminal Management
Data Base Management
Session Services
Peripheral Management
Security
Recovery Management
Network Management

Following is an overview of the modules and the basic services they perform. The major modules and their relationship to other modules will be discussed in detail in Chapters III and IV.

1. Local Communications

- Bus arbitration (traffic management)
- Message transmission and reception including buffer management
- Message control (error detection, correction and acknowledgement)
- Administration (including message accounting, handling of lost or misdirected messages and LAN shutdown)

2. National Communications

- Conversion of the Defense Data Network protocols to LAN protocols and the reverse
- Message assembly and disassembly

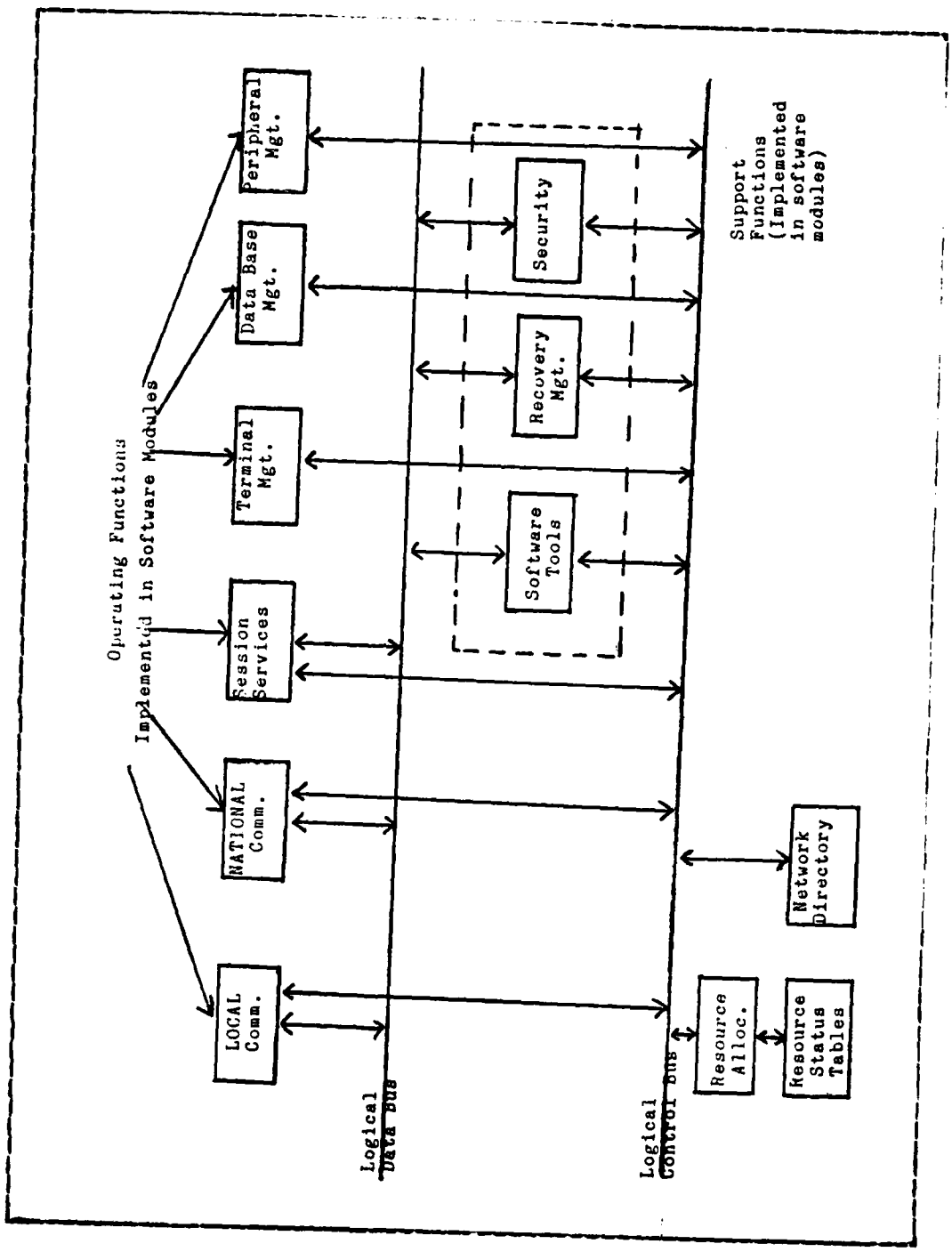


Figure 2.1 Local Network Logical Connections.

3. Front-End Processing

- Terminal and communication line buffering
- Code Conversion
- Byte/word assembly and disassembly
- Control message processing
- Authentication

4. Terminal Management

- Message editing
- Screen management
- Virtual terminal operations

5. Data Base Management

- File creation and updating
- Query processing and data retrieval
- Data dictionary creation and maintenance
- File catalog creation and maintenance

6. Session Services

- Establish and maintain local and remote sessions:

    a. Within the LAN (SPLICE minicomputer processes)

    b. With local host(s) (the mainframe processes)

    c. With remote host(s) (also mainframe processes)

- Provide logical and physical network addresses based on value of a Services Request Code
- Access control

7. Peripheral Management

- Management of Unit Record Input/Output (to include reading cards, printing lines, and spooling files for input and output)
- Optical Character Recognition or Mark Sense Equipment

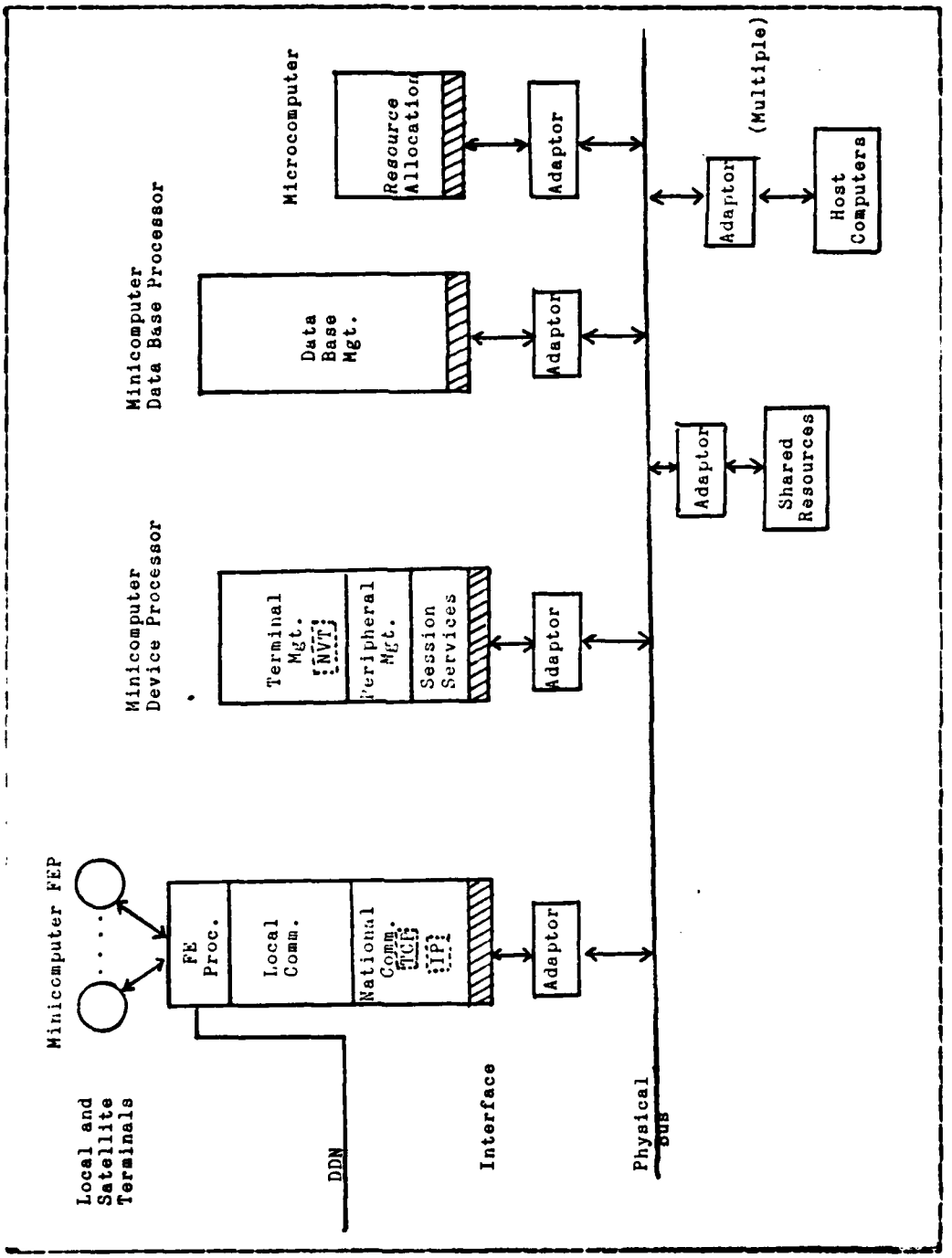


Figure 2.2 Local Network Physical Connection.

8. Security

- Provide positive user identification via passwords
- Control authorization verification
- Provide for reconstruction of critical data
- Provide auditing facilities

9. Recovery Management

- Receive and react to notification of transmission errors (existence of an error condition)
- Maintain LAN copy of Network Directory and update it when physical address changes occur
- Notify the Network Management module and all functional modules and processors within the LAN of changes in LAN status.

10. Network Management

- Perform monitoring and measuring of LAN performance
- Evaluate network performance and identify down or failing components
- Monitor and control LAN interface to long-haul network (DDN)
- Regulate flow of packets between networks and perform other tasks to support this flow
- Provide local failure notification to nodes and hosts outside of LAN and provide LAN with information about outside nodes and hosts
- Provide accounting capabilities (i.e., resource utilization)

III. COMMUNICATIONS FUNCTIONAL MODULES

A. NATIONAL COMMUNICATION (NC) MODULE

It is a common user requirement that a single terminal and access port should be able to access any computing resource that the user may desire -- even if the resource is on another data network. Based on this requirement, there arises a user need to have data networks connected together. Although from user viewpoints the requirement for interconnection is independent of the network technology, this is not true from the implementation viewpoints. There are some considerable complications in connecting networks of relatively different technologies. These interconnections can be viewed primarily in terms of interfaces and network services [Ref. 11].

These can both be divided on the basis of the characteristics they possess - those of datagram or those of virtual circuit. It is important to distinguish datagram and virtual circuit services from datagram and virtual circuit interfaces.

1. Datagram/Virtual Circuit

A datagram interface allows the subscriber to enter packets into the network independent of any other packets which have been or will be entered. Each packet is handled separately by the network. A virtual circuit interface requires an exchange of control information between the subscriber and the network for the purpose of setting up address translation tables, setting up routes or preallocating resources, before any data packets are carried to the destination. Thus, an end-to-end logic circuit must be established.

A datagram service is one in which each packet is accepted and treated by the network independently of all others. Sequenced delivery is not guaranteed. In fact, there is no guarantee that all datagrams will be delivered. Since packets may be independently routed over alternate network paths, duplicate copies of datagrams might be delivered. A virtual circuit service tries to guarantee the sequenced delivery of the packets associated with the same virtual circuit. It typically provides the host with advice from the network on flow control per virtual circuit as opposed to the packet-by-packet acceptance or rejection typical of a datagram service. Any duplicate packets produced are filtered by the destination packet switch before delivery to the subscriber.

2. Operation

The NC module of the local network provides the interface between the LAN and the DDN. In the NC module, both sides of this interface will provide a virtual circuit service [Ref. 4]. Packets will be transmitted on the DDN backbone, while fragments will be transmitted on the LAN. It will be the responsibility of the NC module to provide the interface between the DDN and each LAN. This will require that a conversion be provided between the LAN protocol and TCP, the protocol utilized by the DDN. (See Appendix C for specifics on TCP.) To avoid connecting all functional modules and nodes directly to the DDN, a gateway will be used. The fundamental role of a gateway is to terminate the internal protocols of each network to which it is attached while, at the same time, provide a common ground across which data from one network can pass into another [Ref. 5]. The NC module will function as the gateway and contain both the TCP and LAN protocols.

Messages from the DDN in the form of packets accumulate at the NC. The message or fragment (if the NC has found it necessary to perform fragmentation on the incoming message) is then sent to the destination module over the LAN. The destination module's logical address and its physical address would have been recorded in the message (see Session Services Module) at the originating LAN [Ref. 10].

The task of broadcasting physical address changes to the various LANs must be accomplished and it is currently envisioned that the Network Management Module located at FMSO will handle this. Additionally, each LAN must keep a copy of the network directory and make all necessary changes to the network physical addresses as they occur. This will be done by the resident Recovery Management Module [Ref. 4].

Physically, the NC resides in the Front-End Processor. It will perform host to host flow control but not alternate routing. Messages will be sent to the nearest Packet Switching Node (PSN) of the DDN on a FIFO basis. Since the LAN speed will feasibly be greater than that of the DDN, the NC buffer could easily fill to capacity. One method of handling this potential problem is to only permit a single message to be unacknowledged at a time [Ref. 4]. In addition, it will necessary to reserve buffer space equal to the maximum size message fragment that could be received. By utilizing these restrictions, message handling will be done in a uniform fashion both intra LAN and inter LAN. Certain problems could result from the first restriction, however, and these will be discussed in the last chapter.

3. TCP

As previously mentioned, TCP is the primary host-to-host protocol in the DDN. An overall description of TCP can be found in Appendix C; however, the key characteristics are reiterated here:

- Host-to-Host Protocol

- Resident in the ISO Transport Layer
- Messages delivered in sequence
- Logical full-duplex connections
- Sequence number assigned to each octet
- Message sequencing and acknowledgements controlled by time outs
- Connection name used to refer to connections after connection is established.
- Precedence and security of messages may be established by users
- Window oriented flow control
- Destination TCP reassembles message segments
- Mandatory that acknowledgements be sent

Only those aspects of the TCP which are necessary to convert messages from LAN to DDN format and vice versa will be implemented in the NC. The TCP commands which will be implemented in the NC are as follows:

* OPEN
 - Active: Begin procedure to synchronize connection
 - Passive: Listen for an incoming signal

(Respective modules will be notified by their local and remote NCs when a connection has been made.)

* SEND
 - send data contained in the indicated user buffer on the connection indicated
* RECEIVE
 - allocate a receiving buffer associated with the specified connection
* CLOSE
 - close the specified connection

(Respective modules will be notified by local and remote NCs that the connection is closed.)

* STATUS
 - obtain status of connection

(This command is not always implemented; however, it would be to the advantage of the SPLICE network to utilize it.)

* ABORT

- causes all pending SENDs and RECEIVEs to be aborted. A RESET message is sent to the remote TCP. Respective modules are notified by their local and remote NCs that the connection has been aborted.

It should be mentioned here that the activity of the TCP can be characterized as responding to events. The events mentioned above fall into the category of user calls. Processing is done by the TCP in response to each of the events that occur. In many cases, the processing required will depend on the state of the connection.

4. Network Layers

For sending and receiving messages on the DDN, all seven network layers as defined in the Reference Model of Open Systems Interconnection (OSI) proposed by the International Standards Organization (ISO) [Ref. 12] will be used. See Table II . The TCP format [Ref. 23] will be provided to the DDN by the NC module whenever communication on the DDN is necessary.

5. Addressing

The TCP uses port identifiers in its header to identify the separate data streams that the TCP may handle. Since port identifiers are selected independently by each operating system, TCP, or user, they might not be unique to each TCP. To provide for unique addressing at each TCP, an internet address identifying the TCP is concatenated with a port identifier to create a socket which will be unique throughout all networks connected together [Ref. 10]. The TCPs are free to associate ports with processes in any manner they choose; however, several basic concepts are

TABLE II
ISO Layers in DDN Communication

Layer	Module
Application	Application process modules
Presentation	Terminal Management
Session	Session Services
Transport	TCP
Network	to be specified by the DDN
Data Link	to be specified by the DDN
Physical	to be specified by the DDN

necessary. It is envisioned that processes may "own" ports and that they can only initiate connections on the ports that they own. A connection is specified in the OPEN call by the local port and foreign (destination end) socket arguments. After the connection has been opened, the TCP supplies a local connection name by which the user refers to the connection in subsequent calls [Ref. 23].

In the SPLICE LAN, the port identifier will correspond to the logical address of a functional module. The network address corresponds to that of a physical processor in which the module resides. This will allow for the flexibility and mobility of modules, since logical addresses do not change, whereas physical addresses are subject to change. Both types of addresses will be necessary to access a particular functional module in the SPLICE network.

B. LOCAL COMMUNICATIONS (LC) MODULE

The LC module will also use a virtual circuit service. It will be possible to set up a virtual circuit between any two modules. The circuit would be implemented by creating tables in the Session Services module at both the receiving and sending ends of the connection [Ref. 10]. The virtual circuit can be established between two functional modules residing in the SPLICE minicomputers and also between a functional module residing in a SPLICE minicomputer and a functional module residing in a main frame.

1. Network Layers

The TCP protocol, as it is currently specified, is more complex than necessary for use in a local area network such as SPLICE. In addition, measurements of the TCP [Ref. 23] indicate that it has very poor throughput compared to a high speed (10Mbits/sec) bus such as the one proposed for the SPLICE local area network, and thus will not provide optimal local communication performance. If the entire TCP were included in LAN communications, many extra functions not needed would be unnecessarily implemented. Thus, the best approach seems to be to utilize a subset of the DDN virtual circuit protocol, which is as close to TCP as possible, but which specifies only those portions needed by the LAN. In this manner, translation between the two protocols is simplified and protocol compatibility is provided without the necessity of designing two protocols [Ref. 5].

Overall, a much simpler format, as shown in Table III, will be used for intra Lan communication than that of the entire ISO model. The LAN will not require the detailed services normally provided by the Transport and Network Layers [Ref. 4]. The Presentation Layer functions will be implemented in the Terminal Management module. It will

TABLE III
ISO Layers in LAN Communication

Layer	Module
Application	Application process modules
Presentation	Terminal Management
Session	Session Services
Data Link	Local Communications
Physical	Local Communications

accept data from the application process and convert it to the designed LAN standard format. It will also accept LAN formatted messages and convert them to the appropriate application process format. A terminal user would be considered an "application process" in this conversion activity.

The end-to-end virtual circuit connections (the logical communication linkage between two functional modules) and the fragmentation of complete messages can be implemented in each of the functional modules, as opposed to having them handled by a Transport Layer [Ref. 4]. In this manner, a subset of the DDN virtual circuit protocol (TCP) would be used, as previously described, and the need for a complete Transport Layer would be eliminated.

2. Addressing

To enable implementation of the architecture which has been described, it will require that logical addresses be assigned to the functional modules which will be contained within the SPLICE minicomputers and to the functional modules which currently exist in the SP and ICP main frames. This last assignment will require the identification of programs or packages in the main framesthat make up a functional module. The identification of resources is a central issue in the development of distributed systems in order to provide location independence and the possibility of having multiple copies of the same functionally named resource within the LAN [Ref. 10]. This location independence should permit end users and applications programs the ability to access and manipulate data regardless of whether it resides locally or at another node on the LAN or at any remote node in the SPLICE network. To support this location transparency, it will be necessary to create and maintain a table which will provide the physical address of a hardware unit when the logical address is given. This table and all necessary maintainance functions associated with it will be the responsibility of Session Services [Ref. 10]. Although a table concept was originally developed for a ring network structure [Ref. 13] it can be simulated under other network architectures, such as Ethernet, which also uses a bus structure [Ref. 10].

3. Message Formats

LAN message formats have been designed by many authors. The basic structure provided here was designed in a previous thesis [Ref. 8], however, additional features have been included to incorporate other functions performed by the network [Ref. 4]. If other functions are required, they

can be included at a later date in much the same manner. When a module requires an acknowledgement, the acknowledgement will be piggybacked onto a data message if data is ready to send to the module. If there is no data to transmit, an ordinary acknowledgement message will be used. Requirements for control must be incorporated into this method. These will allow for priority message notification as well as distinguish between new messages and acknowledged messages. It is planned that messages will be transmitted in one continuous stream of bits [Ref. 4]. Although this will simplify the communications protocol, buffer space must be reserved for the maximum size message. To handle long messages which could exceed the maximum buffer size of a functional module, fragmentation is used [Ref. 14]. A fragment is merely a part of a message. In this instance, identification numbers must be provided for both messages and for fragments. Following are illustrations of the intra LAN message formats and descriptions of the packet format fields. The data field length is allowed to vary. All other fields should be fixed length, however specific lengths for these fields can be determined after detailed network configurations and hardware specifications have been established [Ref. 8].

a. Flag

The flag field is a bit pattern which signifies the beginning and ending of a message or message fragment. The beginning flag field is also used to synchronize the receiving processsor with the incoming bit stream. This pattern should be chosen such that its length is sufficient for positive identification and its distortion due to collision with another transmitter's flag is easily discernable by collision detection devices. The use of the flag sequence is data or control information that occurs between flags must be prevented through use of a bit stuffing mechanism.

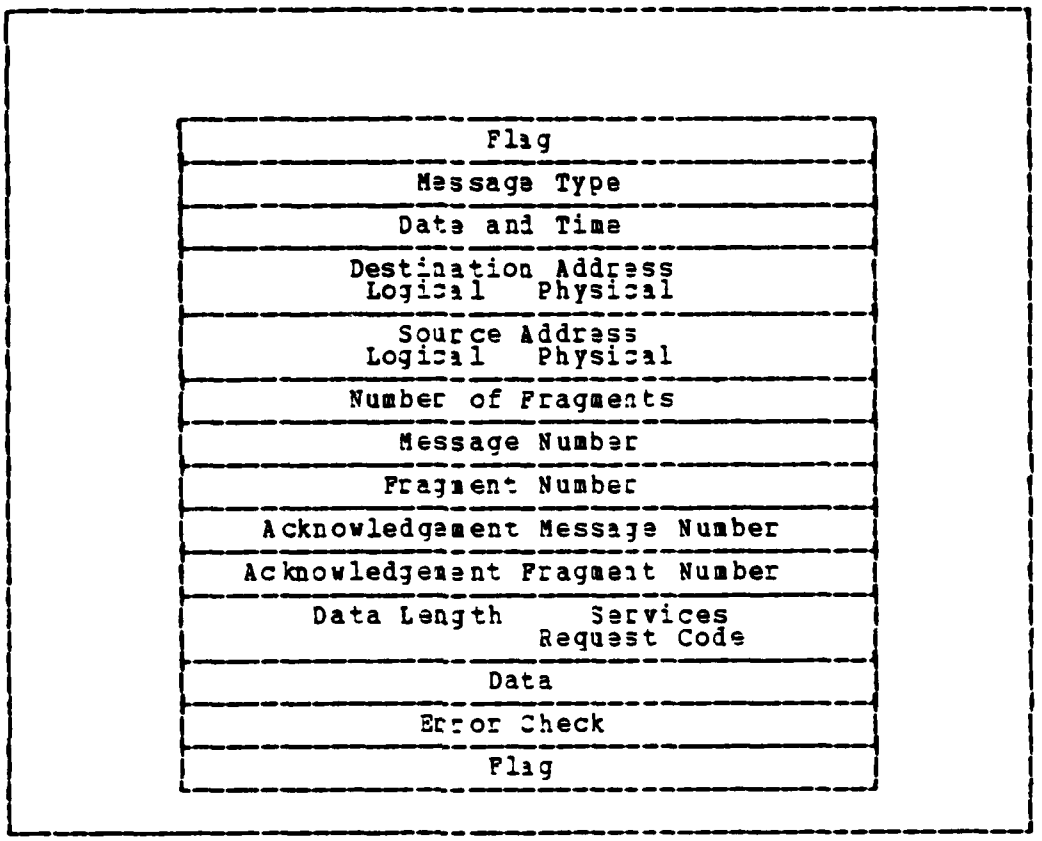


Figure 3.1 LAN Message Format.

b. Message Type

This is a code indicating the type of message being transmitted. The major message types are as follows:

- Normal Data
- Priority Data
- Ordinary Acknowledgement
- Data with Piggybacked Acknowledgement
- Reset LAN (resets communications after error condition)
- Reset Message and Fragment Counts (resets counters to zero)
- LAN shutdown

- others

c. Date and Time

This provides the day, month, year and 24-hour clock time of message transmission from sender.

d. Destination Address

This provides the logical and physical address of the receiving module. It will inform the correct module to copy the rest of the bit stream and continue processing.

e. Source Address

This provides the logical and physical address of the sending module. It is required for proper addressing of acknowledgements and communications control information which must be maintained.

f. Number of Fragments

This provides the number of fragments contained in a message. It is used primarily for message sequencing and for acknowledgement control and also by the receiving module for allocating buffer space.

g. Message Number

This is the sequential number assigned to each transmitted message. If the message being sent is an acknowledgement, the number of the message being acknowledged would be placed in this field. This number is reset to zero on a periodic basis. Each module will be responsible for setting, resetting and incrementing this count. In this manner, the receiver will always know which message it should next receive and the sender will know which message number should next be acknowleiged by the receiver.

h. Fragment Number

This is a sequential number which is assigned to each fragment that belongs to a message. This number will be reset to zero by the sender as soon as the first fragment of a new message is ready to be transmitted. Each module is responsible for setting, resetting and incrementing this count. In this manner, the receiver will always know which fragment number it should next receive. (The message number will be increased after all fragments have been received.) The sender will know which fragment number should be next acknowledged by the receiver. The first fragment will be numbered zero.

i. Acknowledgement Message Number

This provides the message number that is being acknowledged.

j. Acknowledgement Fragment Number

This provides the fragment number that is being acknowledged.

k. Data Length

This field provides the number of bytes in the data portion of a message or message fragment. If no data is being sent, this field can have a value of zero.

l. Services Request Code

This code will indicate which service is being requested by a process. (e.g. retrieve record, update record, etc.)

m. Data

The data field contains the information to be delivered by the network. All other portions of the message header are stripped from the message when it arrives at the receiving module.

n. Error Check

This field contains the Cyclic Redundancy Code (CRC) which is a checksum computed by the sender. If an error is detected by the receiver, the fragment is discarded by the receiver and will not be acknowledged. A match indicates that there are no errors and the message will be accepted by the receiver and acknowledged. If sender does not receive an acknowledgement after an appropriate amount of time, it will assume non-receipt and retransmit. If the second attempt at transmission fails, the Recovery Management module will be notified of an error condition.

It should be noted that the Message Number and Fragment Number fields will be used for data messages and for non-piggybacked acknowledgement messages. The Acknowledgement Message Number and the Acknowledgement Fragment Number fields are only used when the acknowledgement is piggybacked onto the data being sent. The Data Length, Services Request Code and Data fields are only used when data is actually being transmitted in a message.

IV. MANAGEMENT AND CONTROL FUNCTIONAL MODULES

A. SESSION SERVICES (SS) MODULE

The SS module provides the overall controlling mechanism among the clients of the LAN functional resources, i.e. the terminal user and other functional resources themselves. Although the original design of Session Services [Ref. 10] makes a distinction between Controlling SS and Non-controlling SS, for the sake of simplicity and to avoid confusion, this thesis will consider Session Services as one entity containing all described characteristics. Regardless of whether a process is based on an interactive application or on an interactive session (via a LAN user's issuance of query language transactions), there must exist a controlling service to communicate and control the requirements of the user process(es) between itself and the functional resources of the LAN. In order to establish communication between the controlling mechanism, the user and the functional modules, a simple request-accept message transfer needs to be performed [Ref. 10]. This service request code in a message, similar to a user request task, is used by SS to obtain the logical and physical addresses of the functional module(s) which will perform the requested service. An acknowledgement, indicating either acceptance or denial of session support, would be sent to the requesting SS module. After this series of events has been established, the user's process can communicate freely through SS without the need to reconstruct another session service. References to currently supported user sessions would be maintained in a table and could be used to identify the validity of client access for service by the functional resource. SS will

invoke the appropriate functional module which then accesses the user message in the Terminal Management buffer. The module returns the required data after interpreting the instructions in the user message. It is the responsibility of Terminal Management to present the data provided by the functional module in the format needed by the user [Ref. 4]. This entire process involves all the ISO layers (as illustrated in Figure III) required to support intra LAN communications.

If certain requests for service from a user process require coordination and control of multiple LAN functional resources, Session Services will ensure that the request is appropriately broken down into its respective component service requests and that they are performed in the correct order. Basically, SS operates through Terminal Management. This situation results from the fact that user process messages reside in Terminal Management during a session and it is the responsibility of Terminal Management to keep the user informed of all progress associated with the processing of his request [Ref. 10]. SS issues the various messages to the functional modules in support of this request.

B. TERMINAL MANAGEMENT (TM) MODULE

The purpose of the TM module is to provide LAN users with the facilities for communicating simultaneously with a large number of processes spread out among various systems. A terminal user might need to communicate with the LAN or with other local area networks (other Stock Points) through the DDN [Ref. 10]. Since the set of functional modules in this design approach each provide the same basic service, the major place where applications are differentiated is on the terminal screen formats.

Terminal handling has always been somewhat of a problem, since, while terminals exhibit rather similar characteristics, they can differ in very significant ways. In a network architecture, the problem is additionally compounded. Each host must support all n kinds of terminals supported by each of the m hosts on the network. Thus each host must potentially be able to support m x n terminals to permit any user to connect to it. As can easily be seen, this is fairly impractical.

Terminal-oriented protocols are designed to reduce this "m x n" problem to a manageable size by establishing conventions for handling all the terminals on the network [Ref. 15]. One such approach to a terminal protocol defines a network virtual terminal (NVT) [Ref. 16]. In this method, the source terminal side of a connection maps the output of its terminal into the NVT format for transmission through the network. At the destination terminal, the NVT format is mapped into its local format for presentation. The user ends, as defined above, could be processes as well as physical terminals. This approach has the advantage of avoiding the delays and efficiencies of attempting to synchronously share a data structure across a network.

The NVT has several shortcomings, however, which must be considered [Ref. 15]. The introduction of new terminal commands or primitives without modifying the protocol means that each new terminal command will require a minimum of six octets (octet = 8 bits) to be represented. Because the protocol is stream-oriented, every octet of the data stream must be scanned to find control sequences. Secondly, for a virtual terminal protocol (VTP) to be successfully used in a general environment, the virtual terminal must be very well defined. Otherwise, programs that use the terminal in more sophisticated applications such as displaying and updating fields on a CRT will not be able to format their output

deterministically without considerable knowledge of the physical terminal being used [Ref. 15]. There is also the possibility that not all physical terminals may be able to service all virtual functions. Additionally, for a VTP to be generally applicable, it must restrict itself to terminal functions.

European investigations into virtual terminal protocols have made two major contributions: (1) a well-defined virtual terminal and (2) the development of a model for attentions or interrupts [Ref. 17]. The VTP defines an basic framework for the virtual terminal, and classes of virtual terminals are defined that correspond to the classes of real terminals available (e.g., scroll mode, paged, data entry, etc.). Each class uses the basic model and adds to it the facilities and structures required. Thus, the use of terminal class avoids requiring that all implementations support the most sophisticated terminal functions and allows the characteristics of a virtual terminal to more closely resemble the real terminal being used [Ref. 17]. The European designs also provide commands for one side (virtual terminal) to request of the other what options and what range of parameters it supports and for the requested side to report what it can support.

There may be some limitations imposed on the choice of a virtual protocol, however, as it is currently planned for the DDN to use an ARPANET Telnet NVT feature [Ref. 22]. In this case, the NVT protocol will be probably needed in the TM module to enable communication with remote processes over the DDN. A Terminal Management generic module has been proposed [Ref. 18] which attempts to provide a methodology for utilizing a VTP other than NVT and still maintain compatibility with the DDN protocol. It is felt that this proposal should be further examined in terms of its suitability for the SPLICE local network.

C. DATA BASE MANAGEMENT (DBM) MODULE

Although the LAN design provides for distributed control, it does not provide for the total distribution of data bases within the LAN. There is, however, distribution between the interactive Data Base Management System (DBMS) in the SPLICE minicomputer and the Burroughs mainframe batch-oriented DBMS [Ref. 4]. Data bases are, of course, distributed over the entire SPLICE network. The data base functions, however, are centralized within each LAN [Ref. 10]. This will enable the local network to maintain the necessary control and integrity of files, as the catalog, data dictionary and indices will be centralized.

As distributed processing systems continue to grow, database management systems will have to undergo changes to be responsive to the special requirements of the distributed environment. This will be most evident in the situation of the local area network [Ref. 19]. Changes will be needed both in the service provided by database management systems and in the service implementation - the way it is packaged and delivered. System resources cannot be dedicated to single functional modules; for a variety of reasons, including the need to utilize common information, they must be shared. Rothnie and others [Ref. 20] believe that the type of architecture provided by SDD-1 is appropriate for activities requiring access to a single pool of information distributed over a wide geographical area. It will permit decentralized processing for reasons of performance, reliability and flexibility of function, and was designed to manage data bases whose storage is distributed over a network of computers.

Lowenthal [Ref. 19] introduces the concept of a file server. This is a special purpose software module that, as a minimum, would coordinate concurrent access to a given file

from multiple requestors. It can also support file sorting, catalog management and index searching. In a DBMS, the file server will handle the entire database as well as files. With adequate fixed disk storage for highly active files and moveable disk storage for less active files, it should be capable of supporting foreground queries and file maintenance requests.

Closely relating to this concept is that of the backend database management system as discussed by Maryanski [Ref. 21]. This system was originally proposed as a solution to the problems of overloaded data processing installations. Database management functions are offloaded from the existing mainframe to an attached minicomputer. Basically, database requests are presented to the DBMS after they originate in an applications program and are transmitted through the interface routines. When the database request has been completed, the resulting data and status information are returned to the backend interface which initiates transmission of the results to the host. This method does have the advantage of freeing a substantial part of the processor's resources; however, it has a number of drawbacks as well. A major drawback is the performance penalty incurred by the introduction of the interface and communication software and the transmission time of the intercomputer link. Additionally, code conversion will be required for mapping character and integer data between different implementations. Both of the concepts presented are worthy of further investigation as to their applicability to the LAN design. A recommendation for a particular type of DBMS has not been made by previous studies [Refs. 4 ,10], since it was felt that stronger emphasis should be placed on the capabilities of a vendor provided DBMS (i.e., integrity, recovery, dictionary, query language, data accessability, security, etc.) rather than on the specific

model selected. Although a complete design for the database functional module was not developed, a number of management tools (for providing organizational support) and some of the basic functions the DBMS should provide have been identified [Refs. 4 ,10] and these are briefly discussed below.

A Data Sub-Language (DSL) consists of a Data Definition Language (DDL) for defining data objects (fields) and a Data Manipulation Language (DML) for the processing of data objects. As identified by the CODASYL group, the main function of DDL is to describe the content and structure of the database schema and subschema. Although all DBMS have a DDL, these can vary in the extent to which complex relationships can be expressed. The complexity and capabilities of the DDL should be worth careful consideration based on its application by Navy Stock Points and Inventory Control Points. The DMLs are used to transfer data to and from the database. They can be accessed by calls from a procedural language. The capabilities of the DML will directly affect the applications programmer. Since the DDL and DML are closely related, the functionality of each generally determines the degree of responsibility of both the data base administrator and the applications programmer.

Database Query Languages (DQLs) are generally interactive in nature. Often called "end-user facilities," DQL facilities provide direct interaction with the database schema and permit search strategies for data retrieval or updates by approved end users of the DBMS. With a user friendly DLQ, users can perform ad hoc queries or can build user command files for repetitive data entry, retrieval and validation. A fully implemented and varied DQL will greatly support the current and future information requirements needed by the Navy Supply System.

Database utilities cover a multitude of areas, from password security to image management software and database tuning utilities. Additional database software for text and graphic displays, audit trail utilities, database development aids, database reloading and reorganizing aids and database sizing and responsiveness aids are also available. These could prove extremely useful in support of LAN requirements.

Data Dictionaries/Directories (DD/D) are vital in a distributed data environment and they may be implemented in a variety of ways. A data dictionary is used to identify and define the data elements contained in the database, and any relationships that may exist between these data elements. It indicates where data resides geographically and what data are replicated. It should tell who owns the data, who are responsible for the accuracy of the data, who update data, and who is authorized to read the data. The data directory may refer to applications programs, input or report documents, or simply job streams, but generally it supports the use of data elements that were identified in the data dictionary.

The DBM module must maintain the catalog of file names and status for files pertaining to the foreground applications. This catalog should include filename, size, physical address of both file and index, location of backup copy, access restrictions and format. The module should also be able to retrieve records for display, change records, delete and insert records and print both records and entire files. When printing a record, the TM module will route the transaction message to the DBM module. This module will locate and retrieve the record and send it to the Peripheral Management (PM) module for printing. If a user desires to print a file, the DBM module will have to open the file for the PM module. The PM module will then access and spool the

file onto its own disk file. It can then print the file without tying up the rest of the LAN operations [Ref. 4].

Capabilities should exist to prevent data integrity problems when multiple processes read or update the same data. The system should also permit the use of high-level database user languages for data retrieval, report formatting and searching for and updating the data.

Although the above suggestions do not cover every aspect of a DBMS components and functions, they do describe the types of facilities and capabilities which must be considered in providing a completely distributed processing system for the LAN.

V. CONCLUSIONS AND RECOMMENDATIONS

It is felt that the overall design of the LAN in terms of the functional modules presented provides a qualitative and useable design effort for a distributed Local Area Network. A number of issues still remain to be addressed, however, to insure that all functions have been completely considered and developed.

- Continued support of organizational needs is paramount in the design of any LAN. If LAN operability can be terminated by the failure of any single node (functional module), than the LAN has the potential to be highly unreliable. Some effort should be expended on providing dependability through a method for duplication of critical data and critical functions throughout the LAN.
- More effort should be devoted to the development of the DBM module and the DBMS. Careful consideration of the interrelationships between the Stock Points and the Inventory Control Points should be made in the selection of any DBMS to support long range organizational objectives.
- Additional research should be performed in the areas of security and in the management of functional shared resources. The provision of security controls that are tamperproof may best be accomplished by designing them into the hardware. - Although the concept of having only one message outstanding at a time between a pair of functional modules on the LAN provides certain advantages (i.e.,reduced buffer size requirements), there can be a significant drawback as well. Since a monumental share of the communications will be occuring between the Front-End Processor and the Terminal Management and Session Services modules due to the physical connections envisioned (see Figure 2.2), it is quite

feasible that this restriction on window size for the various modules could create a backlog of messages for the bus. It is recommended that additional study be done to determine whether or not it would be more efficient and productive to increase the suggested window size and allow more than one unacknowledged message to exist at a given time.

- Backup and recovery are very important in support of the SPLICE requirement. Additional study should be performed in these areas and a working Recovery Management module should be developed to handle error detection within the LAN.

A Security Management module should be developed after the appropriate appropriate risk analysis has been performed to provide for the important considerations of security and privacy needed in a distributed system.

- A design for a Front-End Processor should be initiated. It is suggested that a programmable front-end processor would be more cost effective in communications control and would provide more flexible solutions to changing communications requirements.

- A menu for dialogues should be incorporated into Session Services to enable users to more easily employ distant databases, making inquiries, searching the data, generating reports, and, where desirable, updating and creating data.

- A Peripheral Management module should be developed to meet the need for unit record input and output control. This functional module will enable users to print lines, have cards read and spool files for input and output. Thorough research should be done to determine what specific requirements may be required to meet current and future needs of the Stock Points and Inventory Control Points.

- A simulation model should be designed to estimate the performance of the LAN as designed. Attention should be particularly focused towards response time and message transit time.

APPENDIX A
ACRONYMS

ACK	Acknowledgement
AM	Application Management
ARPANET	Advanced Research Projects Agency Network
BBN	Bolt Baranek and Newman
CRC	Cyclical Redundancy Check
CRT	Cathode Ray Tube
CM	Communications Management
DARPA	Defense Advanced Research Projects Agency
DBM	Data Base Management
DBMS	Data Base Management System
DD/D	Data Dictionary/Directory
DDL	Data Definition Language
DM	Data Management
DML	Data Manipulation Language
DOD	Department of Defense
DSL	Data Sub-Language
FD	Functional Description
FEP	Front-end Processor
FMSO	Fleet Material Support Office

IMP	Interface Message Processor
IP	Internet Protocol
IPLI	Internet Private Line Interface
LC	Local Communications
NAVSUP	Navy Supply Systems Command
NC	National Communications
NM	Network Management
NVT	Network Virtual Terminal
PM	Peripheral Management
RM	Recovery Management
SM	Security Management
SP	Stock Point
SPLICE	Stock Point Logistics Integrated Communications Environment
SS	System Specification
TAC	Terminal Access
TAPS	Terminal Application Processing System
TCP	Transmission Control Protocol
UADPS-SP	Uniform Automated Data Processing System - Stock Point
VTP	Virtual Terminal Protocol
WIN	WWMCCS Intercomputer Network
WWMCCS	World Wide Military Command and Control System

APPENDIX B
DEFENSE DATA NETWORK I

The following information is provided as a short summary of the Defense Data Network (DDN). The source document used is the Defense Data Network Program Plan revised May 1982 [Ref. 22].

A. GENERAL DESCRIPTION

The DDN is designed to be a single integrated packet-switching data network which meets DOD data network requirements, both present and planned. The DDN takes full advantage of existing operational networks, such as the WWMCCS Intercomputer Network (WIN) and the Advanced Research Projects Agency Network (ARPANET), for hardware, software, operations and maintenance procedures adaptation. Its design will be based primarily on ARPANET technology.

To reduce development, maintenance and logistical support costs, it is planned to standardize components to the maximum extent possible. These components are switching node hardware, switching node software, cryptographic devices, mini-TACs, host front-end devices, host interface devices and multiplexers.

The switching node is a Bolt Baranek and Newman (BBN) C/30, which is a microprogrammed minicomputer that can include TEMPEST/HEMP protection. This is the most current generation of packet switching hardware using the Interface Message Processor (IMP) software. The C/30 is designed for unattended operation and requires no dedicated personnel. DDN will have 171 switching nodes located at approximately 85 geographically distributed sites. These nodes will

reside on military facilities and are secure to a minimum level of SECRET. The DDN will contain a number of network Monitoring Centers (MCs): a principle System MC, an alternate MC, regional MCs in the Pacific and in Europe, and MCs at each keyed community. The MCs monitor the network status, provide for fault isolation and diagnosis, support software maintenance in the nodes and mini-TACs, and maintain network elements information.

The network is designed to minimize communications errors through the use of error detection and correction mechanisms. A Cyclical Redundancy Check (CRC) of 16 bits is associated with host messages on the access line and with packets on trunks to handle burst errors that typically occur. In addition, 16-bit checksums are provided on an end-to-end basis within the switch subnetwork and on a user-to-user basis via the Transmission Control Protocol (TCP). The protection mechanisms used in the switches are error detection and correction hardware to protect against memory failure and checksumming of critical data structures and portions of code.

An availability of at least 99% will be provided by the network to any pair of single-homed users that wish to communicate with each other. Users will have the capability to enhance their availability either by dual access (two access lines to the same switching node) or by dual-homing (a single access line to two switching nodes). The latter method provides an increased network availability of 99.95% and will be used for critical subscribers.

Originating hosts and terminals can assign traffic precedence levels which will then be used by the switching nodes and mini-TACs as a criterion in resource allocation. The switching nodes provide four levels of precedence, preemption of lower precedence connections, non-blocking of host input and reservation of buffers and other traffic

related data structures. The mini-TAC provides four levels of precedence, preemption of input TCP segments and reservation of input buffers. Category I (Flash and Flash Override) traffic has the highest precedence level and will be processed in a non-blocking mode exclusive of all other traffic modes and volumes.

A number of features are provided by DDN to ensure its survivability. All DDN hardware will have HEMP protection in the form of EM shielding, line isolation and surge arresting protection. Uninterruptible power supplies will be provided to those selected sites having no backup power. To facilitate system reconstitution, there will be five mobile reconstitution nodes equipped with MC capabilities. DDN utilizes a dynamically adaptive routing algorithm to automatically route traffic around congested, damaged or destroyed switches and trunks so the system can continue to function. There is also a dense trunking grid to provide redundancy at all possible points in the network.

B. SECURITY AND PRIVACY MEASURES

1. Link Encryption

The KG-84 crypto device is used on all backbone trunks, on all access lines to classified hosts and mini-TACs, and on access lines to sites that act as MCs for the unclassified community. The link encryption provides full period traffic flow security protection by concealing traffic patterns of interswitch traffic, and by concealing subnet monitoring reports, which could reveal traffic analysis information. It will also protect MC-switch control traffic from disclosure. Traffic which is sent from one remote host to another remote host is encrypted first by the Internet Private Line Interface (IPLI) and again by the KG-84 (see Figure B.1).

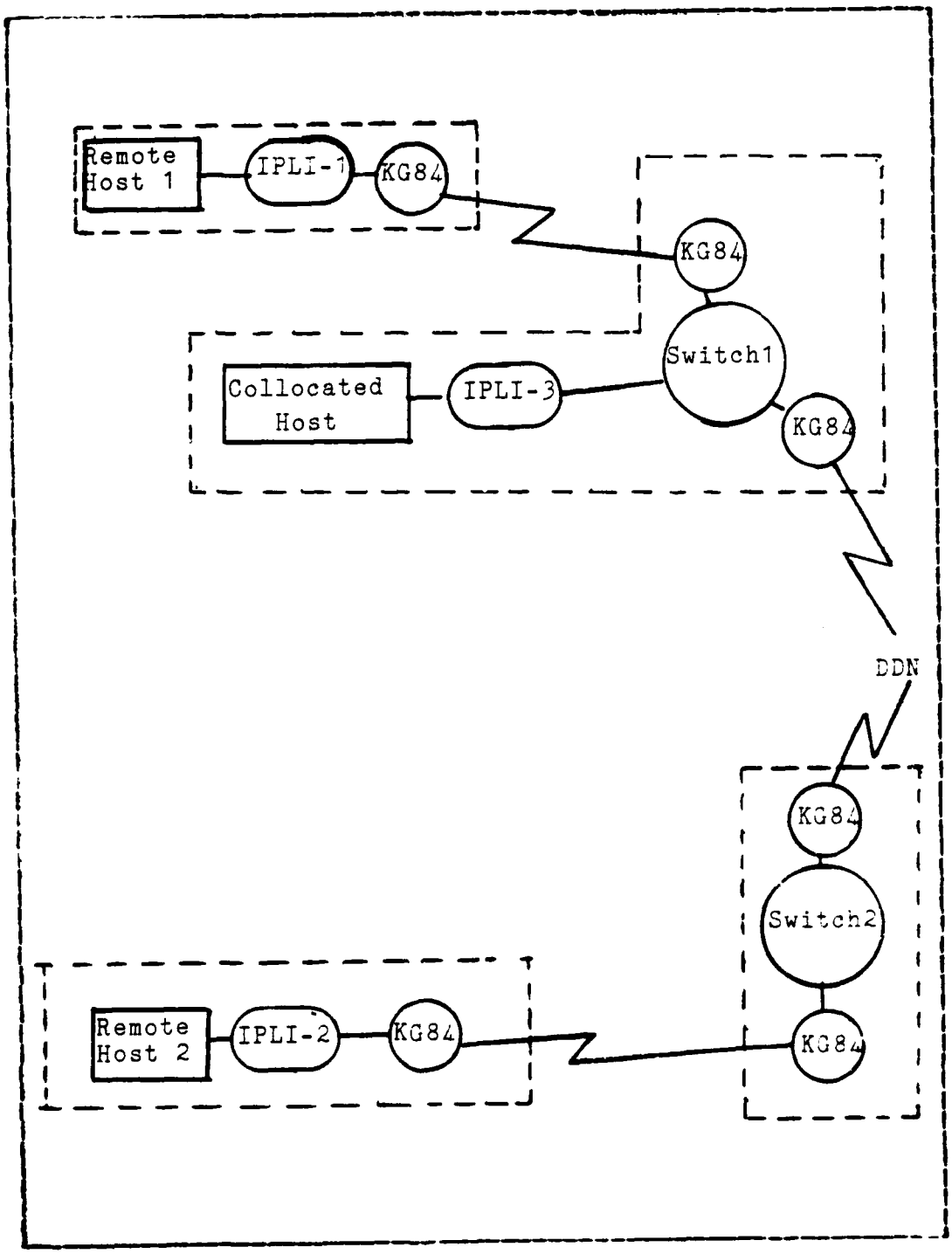


Figure B.1 End-to-End Encryption.

2. Security Level Separation

All DDN subscribers will operate at a specified system high security level. Separation of subscribers at different levels is provided by the use of IPLIs. Thus there will be at least on IPLI key for each different system high level, and, at minimum, one such logical subnet for each security level. The IP and subnet headers must be in the clear for packet processing within the switch, yet they provide information about subscriber traffic patterns and traffic analysis information is classified secret. This problem is handled by link encryption of subscriber access lines and by ensuring that all switching nodes are TEMPEST enclosed and in secure military facilities. To prevent misdelivery of traffic statistics by the subnet, each MC and the "fake" host in each switch that communicates with the MC will be members of a logical subnet that includes only these members. The requests from the MC that trigger collection and reporting of traffic statistics will be protected using a cryptographic authentication protocol.

3. Separation of Communities of Interest

Communities of interest are subscriber groups which 1) present an acceptable level of risk to each other and 2) require a high level of interoperability. Separation of communities of interest is accomplished through the creation of logical subnets by cryptographic means, by software control, or both. For unclassified subscribers, the switches provide the ability to define logical subnets which confine traffic flow only to the members of that logical subnet. These logical subnets are established by the SMC. Currently the switches allow for up to 16 such subnets, but this can be easily increased to 32 or 64.

Rigorous separation for classified user communities is provided by IPLIs. As mentioned earlier, there will be at least one IPLI subnet (collection of like-keyed IPLIs) for each security level. At this time, a community of interest is limited by policy to 128 subscribers.

4. Individual Access Control

Access control to subscribers facilities is the responsibility of the subscribers themselves. The network will assure, based on a number of special mechanisms, that the access of one subscriber to another is controlled with respect to authorized security level and community of interest. Network facilities do not verify, however, that an individual user (person or process) attempting to access a subscriber has valid access rights to that subscriber. Access control to mini-TACs is provided by physical access control measures, as mini-TAC access is only available through hardwire lines.

5. Personnel Clearance Requirements

All personnel with access to switches must be cleared to secret level due to the traffic analysis potential. This clearance level also applies to all personnel at the SMCs and RMCs. Personnel manning a MC for a secure subnet must be cleared to the level of the subnet subscribers, allowing personnel access to the corresponding IPLIs. Crypto technicians will be needed for keying the IPLIs for each community and for link KGs. The keying material for each IPLI community is available only at the IPLI sites. The keying material for the link KGs is available on a pairwise basis at the switch sites based on switch connectivity.

C. MAJOR HARDWARE ELEMENTS

1. Switching Node

The switching node is a BBN C/30 packet switching processor in a TEMPEST/HEMP package. The C/30 hardware is a multi-board, microprogrammed minicomputer with 64K words of Random Access Memory (RAM). It supports a full range of synchronous and asynchronous I/O interfaces. The C/30 software is the ARPANET Interface Message Processor (IMP) program. IMP software can be loaded locally (from a cassette) or by a downline load under MC control. The software provides four major functional capabilities: 1) Tandem (store and forward) traffic processing 2) Host access and end-to end traffic processing, using a variety of host access protocols 3) Routing via a dynamic, adaptive distributed routing algorithm that measures actual packet delays and routes individual packets along the least delay path 4) Monitoring and control services.

2. Internet Private Line Interface

The IPLI is a security device, currently under development, which supports the DOD standard IP protocol and provides end-to-end encryption. IPLI is composed of three functional units: a KG 84 cryptographic device and two MC68000 based packet processors (one on the red side and one on the black side of the KG 84). Two hardware interfaces are provided on each side of the IPLI. Initially used for backup purposes, they will later provide for dual-homing topologies.

The software in each processor will be based on the CMOS operating system being used for a variety of packet-switching applications. It will have the basic functions necessary for the DOD standard internet environment and for monitoring and control. TCP and other protocols which exist

above the IMP can be supported since the IPLI has no knowledge of the TCP and packet processing occurring at the Internet Protocol lower level.

3. Mini-TAC

A mini-TAC is a terminal access device that allows a cluster of up to 16 synchronous and asynchronous terminals to access the network. It is logically equivalent to a network host, particularly in that it uses the same host-host protocols. A mini-TAC will be constructed around a Motorola MC68000 microprocessor with memory, 16 synchronous or asynchronous terminal ports and multiple network interface ports. The mini-TAC will meet TEMPEST and HEMP requirements.

The mini-TAC software allows terminal users to establish connections between their terminals and an arbitrary host on the network. The software will multiplex all of the terminal-host connections over a single TAC-IMP access line. Mini-TACs will communicate with other network hosts using the DOD standard TCP and IP. The Telnet protocol will be used to provide terminal level support.

The mini-TACs will be designed for unattended operation and will require no dedicated personnel. All control functions and hardware and software fault diagnosis can be done remotely from the network Monitoring Centers.

APPENDIX C
TRANSMISSION CONTROL PROTOCOL

This appendix presents a simple description of the Transmission Control Protocol. The information provided was obtained from the DARPA Transmission Control Protocol document of January 1980 [Ref. 23].

A. GENERAL

The Transmission Control Protocol (TCP) is intended for use as a highly reliable standard for the transmission and reception of messages between host computers in a packet switched computer communications environment. This internetwork environment consists of hosts connected to networks which are in turn interconnected via gateways.

TCP is a connection-oriented, end-to-end reliable protocol designed to fit into a layered hierarchy of protocols which support multi network applications. It provides for reliable inter-process communication between pairs of processes in host computers attached to distinct but interconnected computer communications networks. The TCP fits into a layered protocol architecture just above a basic Internet Protocol (IP) which provides a way for the TCP to send and receive blocks of data, called datagrams, through multiple networks and interconnecting gateways. (See Figure C.1).

B. BASIC FUNCTIONS

Since the primary purpose of TCP is to provide reliable, securable and logical circuit or connection service between pairs of processes the following facilities are required:

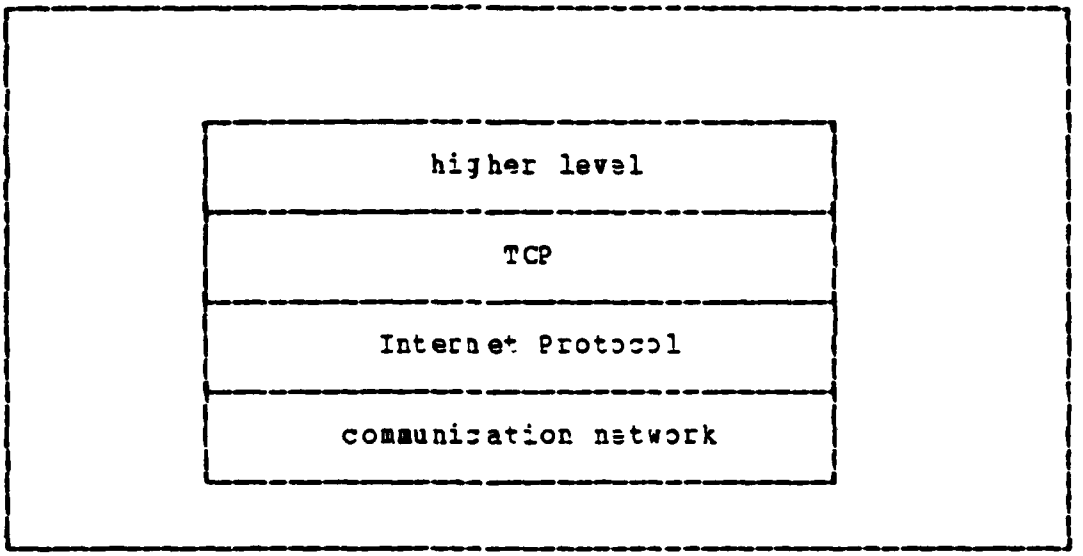


Figure C.1 Protocol Layering.

1. Basic Data Transfer

By packaging some number of octets into segments for transmission through the internet system, the TCP is able to transfer a continuous stream of octets in each direction between its users. The decision to block or forward data is made by the TCP at its own convenience. Users are also allowed to submit records, called letters, for transmission. In this case, the TCPs will forward and deliver data up to the record boundary (end-of-letter) that was specified by the sending user.

2. Reliability

The TCP must be able to recover when data is damaged, lost, duplicated or delivered out of order by the internet communications system. To accomplish this, each octet transmitted is assigned a sequence number, and a positive acknowledgement (ACK) is required from the receiving

TCP within a specified time interval. If the ACK is not received, data is retransmitted.

The sequence numbers are used by the receiver to correctly order segments arriving out of order and to eliminate duplicates. A checksum is added to each segment transmitted to deal with damage occurrance. This checksum is verified at receiving end and all damaged segments are discarded. If the internet system does not become completely partitioned, properly functioning TCPs will be able to recover from internet communication system errors.

3. Flow Control

The receiver can govern the amount of data sent by the sender by returning a "window" with every ACK indicating a range of acceptable sequence numbers beyond that of the last successfully received segment. In stream mode, the window indicates the number of allowable octets the sender may transmit before further permission must be obtained. In record mode, the window tells the allowed amount of buffer space the sender may consume.

4. Multiplexing

The TCP provides a set of addresses or ports within each host to permit many processes within that host to use TCP communication facilities simultaneously. These are concatenated with the network and host addresses from the internet communication layer and form a socket. Each connection is uniquely identified by a pair of sockets, so one socket may be used in multiple connections.

The connection of ports to processes is independently handled by each host, however it proves useful to attach frequently used processes to fixed sockets and make these addresses known to users. These services can then be accessed more easily.

5. Connections

TCPs are required by both the reliability and flow control mechanisms to initialize and maintain certain status information for each data stream. The combination of this information, which includes sockets, sequence numbers and window sizes, is called a connection. The pair of sockets that identifies its two sides uniquely specifies a connection.

If two processes wish to communicate, their TCPs must first establish a connection. This is accomplished through the initialization of the status information on each side. When the communication is complete, the connection is terminated or closed to free the resources for other users. Since these connections must be established between hosts over a somewhat unreliable internet communication system, a handshake mechanism with clock-based sequence numbers is used to avoid erroneous initialization of connections.

6. Precedence and Security

The users of TCP may indicate the security and precedence of their communication. Since not all TCP modules will necessarily function in a multilevel secure environment, some may be limited to unclassified use only and others may operate at only one security level. Provision is made for default values to be used when these features are not needed.

C. MODEL OF OPERATION

Processes transmit data by calling on the TCP and passing buffers of data as arguments. The TCP packages the data from these buffers into segments and calls on the internet module to transmit each segment to the destination TCP. The receiving TCP places the data from a segment into

the receiving user's buffer and notifies the receiving user. Control information is included in the segments used by the TCPs to insure reliable ordered data transmission.

Each TCP has an internet protocol module associated with it to provide an interface to the local network. This internet module packages TCP segments inside internet datagrams and routes these datagrams to a destination internet module or intermediate gateway. To transmit the datagram through the local network, it is embedded in a local network packet. The packet switches may perform further packaging, fragmentation or other operations to achieve the delivery of the local packet to the destination internet module.

At a gateway between networks, the internet datagram is "unwrapped" from its local packet and examined to determine through which network the internet datagram will travel next. The internet datagram is then "rewrapped" in a local packet suitable to the next network and routed to the next gateway, or to the final destination.

A gateway can break up an internet datagram into smaller internet datagram fragments if needed to allow transmission through the next network. To accomplish this, the gateway produces a set of internet datagrams, each of which contains a fragment of the original. These fragments may be broken down again at intermediate gateways. The fragment format is designed so that the destination internet module can reassemble these fragments into internet datagrams. A destination module unwraps the segment from the datagram (after fragments have been reassembled, if necessary) and passes it to the destination TCP.

It should be noted that the mechanisms of TCP do not preclude implementation of the TCP in a front-end processor, however, in such a case a host-to-front-end protocol must provide the functionality to support the type of TCP-user interface required.

LIST OF REFERENCES

1. U.S. Navy Fleet Material Support Office, Environment Division: Code 9441, Stock Point Logistics Integrated Communications Environment (SPLICE), Software Design, 19 March 1979.

2. Fleet Material Support Office, Department of the Navy, Document No. F94L0-001-9260-SS-SU01, Stock Point Logistics Integrated Communications Environment (SPLICE), System Specification, 2 February 1981.

3. Fleet Material Support Office, Department of the Navy, Document No. F94L0-001-9260-FD-SU01, Stock Point Logistics Integrated Communications Environment (SPLICE), Functional Description, 1 May 1980.

4. Naval Postgraduate School Report NPS-54-82-003, Functional Design of a Local Area Network for the Stock Point Logistics Integrated Communications Environment, by Dr. N. F. Schneidewind, December 1982.

5. Clark, D.D. et. al., "An Introduction to Local Area Networks," Proceedings of the IEEE, Vol 66, No 11, pp. 1497-1517 November 1978.

6. Metcalf, R. and Boggs, D., "Ethernet: Distributed Packet Switching for Local Computer Networks," Communications of the ACM, Vol. 19, No. 7, pp. 395-404, July 1976.

7. Gordon, R.L., Farr, W.W., and Levine, P., "Ringnet: A Packet Switched Local Network with Decentralized Control," Proceedings of the Local Area Network Symposium, pp. 13-19, May 1979.

8. Inman, K.A. Jr., and Marthouse, R.C., "Supply Point Logistics Integrated Supply Point Logistics Integrated Communications Environment (SPLICE) Local Area Computer Network Design Issues for Communications," Master's Thesis, Naval Postgraduate School, Monterey, California, June 1982.

9. Luczak, E.C., "Global Bus Computer Communications Techniques," Computer Networking Symposium, pp. 58-71, December 1978.

10. Reinhart, J.N. III and Arana, R., Database and Terminal Management Functional Design Specifications in Support of Stock Point Logistics Integrated Communications Environment (SPLICE) Master's Thesis, Naval Postgraduate School, Monterey, California, June 1982.

11. Cerf, V.G. and Kirstein, P.T., "Issues in Packet-Network Interconnection," Proceedings of the IEEE, Vol. 66, No. 11, pp. 1385-1408, November 1978.

12. Tanenbaum, A.S., Computer Networks, Prentice-Hall, Inc., 1981.

13. Lunn, K. and Bennett, K.H., "An Algorithm for Resource Location in a Loosly Linked Distributed Computer System," Association for Computing Machinery, Operating System Review, Vol 15, No. 2, pp. 16-20, April 1981.

14. Sunshine, C. A., "Transport Protocols for Computer Networks," in Kuo, F.F., (ed.), Protocols and Techniques for Data Communication Networks, Prentice-Hall, pp. 35-74, 1981.

15. Day, J. D.,"Terminal Protocols," IEEE Transactions on Communications, Vol 18, No. 4, pp. 604-611, April 1980.

16. Davidson, J.,and others, "The ARPANET Telnet Protocol: Its Purpose, Principles, Implementation, and Impact on Host Operating System Design," Proceedings of the 5th Data Communications Symposium, 1977.

17. Day, J. D., "Terminal, File Transfer, and Remote Job Protocols for Heterogeneous Computer Networks," in Kuo, F. F., (ed.), Protocols and Techniques for Data Communication Networks, Prentice-Hall, pp. 78-121, 1981.

18. Barnes, J. D., Local Area Network Terminal Management in Support of Supply Point Logistics Integrated Communications Environment (SPLICE), Master's Thesis, Naval Postgraduate School, Monterey, California, December 1982.

19. Lowenthal, E., "Database Systems for Local Nets," Datamation, Vol 28, No. 9, pp. 96-106, August 1982.

20. Rothnie, J.B., Jr., and others, "Introduction to a System for Distributed Data Bases (SDD-1)," ACM Transactions on Database Systems, Vol 5, No. 1, March 1980.

21. Maryanski, F. J., "Backend Databae Systems," ACM Computing Surveys, Vol 12, No. 1, pp. 3-25, March 1980.

22. Defense Communications Agency, Defense Data Network Program Plan, revised May 1982

23. University of Southern California, Information Sciences Institute, Defense Advanced Research Projects Agency, Transmission Control Protocol, January 1980

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virgina 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 59 Department of Administrative Sciences Naval Postgraduate School Monterey, California 93940	1
4.	Curricular Office, Code 37 Computer Technology Naval Postgraduate School Monterey, California 93940	1
5.	Professor Norman F. Schneidewind, Code 54S Department of Computer Science Naval Postgraduate School Monterey, California 93940	2
6.	Professor N. R. Lyons, Code 54LB Department of Administrative Sciences Naval Postgraduate School Monterey, California 93940	1
7.	Lieutenant Commander Kathleen Barrett, USN Naval Data Automation Center, Code 40 Washington Navy Yard Washington, D.C. 20374	1
8.	Ms Mary Willoughby P.O. Box 94 Mendocino, California 95460	1
9.	LCDR Dana Fuller, USN Commander, Naval Supply Systems Command Code 0415A Washington, D.C. 20379	1
10.	LCDR Ted Case, USN Fleet Material Support Office Code 94L Mechanicsburg, Pennsylvania 17055	1


DAT
FILM
5-8